PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2002-312167

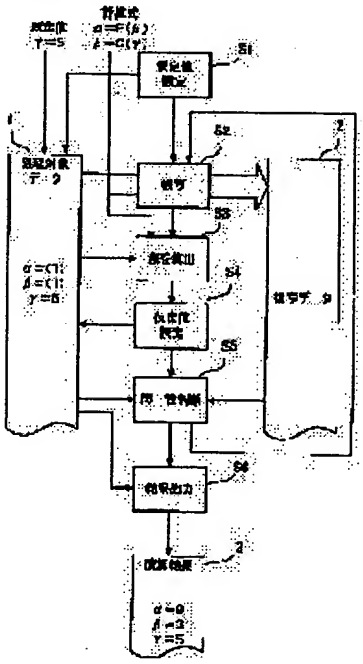
(43)Date of publication of application : 25.10.2002

(51)Int.Cl. G06F 9/44
G06F 9/45
G06F 17/12

(21)Application number : 2001-114715 (71)Applicant : FUJITSU LTD

(22)Date of filing : 13.04.2001 (72)Inventor : MATSUDA CHIAKI

(54) PROGRAM FOR MAKING COMPUTER CALCULATE VALUE OF VARIABLE, COMPILE PROGRAM, VARIABLE VALUE DETERMINING METHOD, AND PROGRAM GENERATING METHOD



(57)Abstract:

PROBLEM TO BE SOLVED: To provide a program capable of optimizing processing efficiency and allowing a computer to calculate the values of variables, irrespective of the sequence of description or the sequence of execution of calculation expressions.

SOLUTION: An assumed value is set for an unknown variable (step S1). Each time an assumptive value is set for the unknown variable, the solutions of the respective calculation expressions are calculated, based on predetermined values of known variables and on the assumptive value set for the unknown variable (step S3). The solution calculated, based on each of the plurality of calculation expressions, is set as a new assumptive value for the unknown variable to be calculated by the calculation expression (step S4). Each time such an assumed value is set for unknown variable, identity is determined between assumptive values for the unknown variable before and

[Claim(s)]

[Claim 1] The program for making a computer compute the value of a variable, characterized by making said computer execute process of;

when two or more calculation expressions to calculate at least one unknown variable and a predetermined value of at least one known variable in said plural calculation expressions are inputted, setting an assumptive value for said unknown variable,

each time an assumptive value is set for said unknown variable, repeatedly calculating the solutions of said respective calculation expressions, based on predetermined values of said known variables and on the assumptive value set for said unknown variable,

setting said solutions of being calculated, based on said each of plural calculation expressions, as assumptive value of said unknown variable calculated by said respective calculation expressions,

each time assumptive value is set for said unknown variable, determining identity between assumptive values for said unknown variable before and after the setting,

when assumptive value for said unknown variable, before and after the setting, are determined as being the same, outputting assumptive value set for said unknown variable as the calculation result.

[Claim 2] The program according to claim 1 characterized by, in the way of determining identity, determining identity positive when all pairs of assumptive values for said unknown variable before and after the setting are same.

[Claim 3] The program according to claim 1 characterized by, in the way of determining identity, determining identity positive when any of all pairs of assumptive values for said unknown variable before and after the setting are same.

[Claim 4] The program according to claim 1 characterized by, in the way of determining identity, determining identity positive when assumptive values for said unknown variable before and after the setting are approximate.

[Claim 5]

The program according to claim 1 characterized by;

before setting assumptive value for said unknown variable, recording assumptive value of said unknown value before the setting,

in the way of determining identity, determining identity positive when any of all pairs of assumptive values for said unknown variable before and after the setting are same.

[Claim 6] The program according to claim 1 characterized by, in calculating a solution of said respective calculation expression, calculating calculation expressions in parallel by using plural of processors.

[Claim 7] The program according to claim 6 characterized by, in calculating a solution of said respective calculation expression, executing calculation, in parallel by using different processors, of plural calculation expressions having no priority of processing order in decision table designating processing order to be executed.

[Claim 8] The compile program for generating the program, where instructions for compute values of variables are described, characterized by making computer, that is inputted plural calculation expressions to compute at least one unknown variable, execute process of;
responding to input of predetermined value of at least one known value in said plural calculation expressions, and generating instructions to set assumptive values for said unknown variables,
generating instructions that, each time an assumptive value is set for said unknown variable, repeatedly calculate the solutions of said respective calculation expressions, based on predetermined values of said known variables and on the assumptive value set for said unknown variable,
generating instructions that set said solutions of being calculated, based on said each of plural calculation expressions, as assumptive value of said unknown variable calculated by said respective calculation expressions,
generating instructions that, each time assumptive value is set for said unknown variable, determine identity between assumptive values for said unknown variable before and after the setting,
generating instructions that, when assumptive value for said unknown variable before and after the setting are determined as being the same, output assumptive value set for said unknown variable as the calculation result.

[Claim 9] The method of determining values of variables by computer, characterized by;

responding to plural calculation expressions to compute at least one unknown variable and predetermined value of at least one known value in said plural calculation expressions, setting assumptive values for said unknown variables,

each time an assumptive value is set for said unknown variable, repeatedly calculating the solutions of said respective calculation expressions, based on predetermined values of said known variables and on the assumptive value set for said unknown variable,

setting said solutions of being calculated, based on said each of plural calculation expressions, as assumptive value of said unknown variable calculated by said respective calculation expressions,

each time assumptive value is set for said unknown variable, determining identity between assumptive values for said unknown variable before and after the setting,

when assumptive value for said unknown variable before and after the setting are determined as being the same, outputting assumptive value set for said unknown variable as the calculation result.

[Claim 10] The method of generating program for generating program for, based on plural calculation expressions to compute at least one unknown variable, calculating values of said unknown variables, characterized by;

responding predetermined value of at least one known value in said plural calculation expressions, and generating instructions for setting assumptive values for said unknown variables,

generating instructions that, each time an assumptive value is set for said unknown variable, repeatedly calculate the solutions of said respective calculation expressions, based on predetermined values of said known variables and on the assumptive value set for said unknown variable,

generating instructions that set said solutions of being calculated, based on said each of plural calculation expressions, as assumptive value of said unknown variable calculated by said respective calculation expressions,

generating instructions that, each time assumptive value is set for said unknown variable, determine identity between assumptive values for said unknown variable before and after the setting,

generating instructions that, when assumptive value for said unknown variable before and after the setting are determined as being the same, output assumptive value set for said unknown variable as the calculation

result.

DETAILED DESCRIPTION

[Detailed Description of the Invention]

[0001]

[Field of the Invention]

This invention relates to the program, the compile program, the variable-value decision approach, and program generation method for making a computer compute values of variables, especially relates to the program, the compile program, the variable-value decision approach, and program generation method for making a computer compute values of variables without considering the execution order of the expressions for computing variables.

[0002]

[Description of the Prior Art]

Computerization of business is attained in many companies currently. Since there are various types of industry in the industrial world, in order to computerize the business of each type of industry, it is necessary to develop the software applied to each business. The program of software can be created using C language, Assembler, etc.

[0003] Such programming is fundamentally performed by human being's manual entry. However, a mistake is inevitable in human actions. Therefore, generally it is thought that a mistake is surely in a program. The mistake of such a program is called the bug. If these bugs are not removed, software cannot attain the expected goal, or, computer operation executed by such program becomes unstable.

[0004] Types of bugs are roughly divided into the following three kinds.

1) The grammatical mistake that is discovered during translation.

2) The logical mistake that is discovered by immediate errors during execution.

3) The logical mistake that do not cause errors during execution but cause mistakes in execution results.

[0005] Usually, if a program is created, bugs are discovered and they are corrected (debugging).

The bugs of type 1) are a mistake of incorrect description of functions, or a mistake in the number of parentheses. The bug by the mistake of such syntax is immediately discovered by the compiler (compiler), and an error message is outputted. Therefore, discovery of the bug by syntax mistake is easy.

[0006] The bugs of type 2) are not incorrect in syntax, but the number of the data handed over between modules are incorrect, or data format are incorrect. Such

bugs are difficult to be discovered by the compiler, and can be discovered by testing partially or by performing a whole program.

[0007] However, the bugs of type 3 is difficult to discover in order that an error message may not come out in a compiler. For example, they are the following expressions to compute a value of the variable a:

[0008]

[Equation 1] Variable a0= variable b0x variable c0 ... (1)

[0009]

[Equation 2] Variable b0= variable c0+ variable d0 ... (2)

[0010]

[Equation 3] Variable c0= variable d0x variable e0 ... (3)

Here, when the value of a variable d0 and a variable e0 is given, it is necessary to create a program so that a computer can compute the expressions in order of the expression (3), the expression (2), and the expression (1). If the order of execution of such expressions is incorrect, the incorrect execution result will be outputted. In this case, since it is grammatically correct, an error message is not given during a translation. Moreover, an error message is not outputted even if it performs. Therefore, in order to discover the bug of type 3, conventionally human detections are given to find mistakes in an execution result.

[0011]

[Problem(s) to be Solved by the Invention]

However, human being needs to check and find out the mistake of an execution result by visual inspection, and it is difficult to discover the type 3 bugs. Therefore, in order to create reliable software, we have to spend the long time on the debugging job to remove bugs from a program. Consequently, completion of software may delay as long as time spent for debugging and productivity of software development declines.

[0012] Then, it is considered to prevent the type 3 bugs. When we consider the causes for the type 3 bugs, it can be said the reasons are peculiar to the von Neumann computer.

[0013] The von Neumann computer is invented when hardware is made of vacuum tub. It was an indispensable technical problem that processing have to be finished in one time in respect of performance or memory space saving, since throughput of the computer of this time was very small comparing with a current computer. Therefore, inevitably, useless repeat processing is not performed, but software design logic which finishes processes in one time is required, and such logic is followed until now. When two or more variables are expressed in arithmetic expressions using four operators, in

order to prevent useless repeats, for example, the arithmetic expressions are to be executed in order from the most subordinate concept.

[0014] Drawing 15 is the drawing showing the example of description and the example of execution of a conventional-type program. The description example 910 of the conventional program and the execution example 920 according to the description are shown in drawing 15 .

[0015] The definition of data etc. and description of procedure have accomplished in the example 910 of description of the program of a conventional type. According to description of possessing order, the program "the parent program ABCD" executing the expressions of superordinate concept calls the child program "the child program AB and the child program CD" which execute the expressions of subordinate concept. Similarly, the grandchild program "the grandchild program A and the grandchild program B" executing the expressions of more subordinate concept is called from the child program (child program AB). The child program (child program CD) calls the grandchild program "the grandchild program C and the grandchild program D" executing the expressions of more subordinate. It returns to the parent program calling subordinates after called programs terminate processing.

[0016] According to the example 920 of program execution, the definition of data etc. is performed at the time of program execution. Then, a program is executed according to processing order of the call relation.

[0017] Thus, in the program of a conventional type, logic (logic) is designed supposing the sequence of execution. If such logic becomes complicated, it will be easy to cause bugs. That is, based on a general requirements definition and a general external design, programmers are forced to create documents based on the logical structure for computers and to do testing job. This is one of the factors of bugs.

[0018] Therefore, in respect of productivity and maintainability, the method of software development according to the von Neumann computer is inefficient, and its development period is hard to be controlled. Thereby, construction development costs are also likely to increase from a planned one. In order to improve the development method of the software fundamentally, eliminating many evil problems, the new concept and new theory replacing with the old approach are quested.

[0019] For example, the technique based on the theory named Lyee (coined by collecting the alphabetic characters of the tail of each word of GOVERNMENTAL METHODOLOGY for SOFTWARE PROVIDENCE) is exhibited in the homepage (http://www.lyee.co.jp/jp/index.htm). According to this technique, a programmer expresses two or more words (variable) by arithmetic expressions using four

operators and so on, and describes them in order from the most superordinate one. Then it is pre-compiled and is made to be executed from the most subordinate one. International patent application (WO97/16784, WO98/19232) relevant to this technique was also filed.

[0020] Thus, generating of the bug by the mistake of the order of description of arithmetic expressions like the type 3 bugs can be prevented by applying the programming technique which does not require consideration of order of expressions in description of a program. It is considered that this method is very useful in software development and application range of its use will expand.

[0021] However, as for the Lyee theory, it is unknown whether it is applicable to all algorithms, and the means of sufficient optimization is not indicated even in view of processing effectiveness of a program. Then, variety of the programming technique to make program without considering order of description or execution and optimization of processing effectiveness are desired.

[0022] As this invention is made in view of such point, optimization of processing effectiveness is possible, and it aims at offering a program, a compile program, the variable-value decision approach, and a program generation method in order to make a computer compute values of variables, without considering order of description or execution.

[0023]

[Means for Solving the Problem]

In this invention, in order to solve the above-mentioned technical problem, the program and the variable-value decision approach for making a computer compute the value of a variable as shown in drawing 1 are offered. By making a computer execute the program of this invention, processing of the following variable-value decision approaches is realized on a computer.

[0024] An input of two or more formulas for computing at least one strange variable and the default of at least one known variable contained in two or more formulas sets up an assumption value to a strange variable (step S1). Whenever an assumption value is set up to a strange variable, the solution of each formula is computed based on the default of a known variable, and the assumption value set as the strange variable (step S3). The solution computed based on two or more formulas of each is newly set up as an assumption value of the strange variable computed in each formula (step S4). Whenever an assumption value is set up to a strange variable, the identity of the value before and behind a setup of the assumption value of a strange variable is judged (step S5). When the value before and behind a setup of the assumption value of a

strange variable is judged to be those with identity, the assumption value set as the strange variable is outputted as the result of an operation 3 (step S6).

[0025] Thereby, an input of the default of a formula and a fixed variable calculates the solution to with the default of a fixed variable, and the assumption value of an undecided variable. By the calculated solution, an assumption value is newly set up to an undecided variable. And if there is no identity in the value in the installation order of an assumption value, calculation of the solution to and a setup of an assumption value will be repeated. If identity is in the value in the setup order of an assumption value, the assumption value set as the strange variable at that time will be outputted as the result of an operation.

[0026] Moreover, it sets to the compile program and program generation method for generating the program the instruction for making the value of a variable compute was described to be, in order to solve the above-mentioned technical problem in this invention. The computer into which two or more formulas for computing at least one strange variable were inputted is answered at the input of the default of at least one known variable contained in said two or more formulas. Whenever it generates the instruction which sets up an assumption value to said strange variable and an assumption value is set up to said strange variable It is based on the default of said known variable, and the assumption value set as said strange variable. Said solution which generated the instruction which repeats and computes the solution of each of said formula, and was computed based on said each of two or more formulas The instruction newly set up as an assumption value of said strange variable computed in said each formula is generated. Whenever an assumption value is set up to said strange variable, the instruction which judges the identity of the value before and behind a setup of the assumption value of said strange variable is generated. When the value before and behind a setup of the assumption value of said strange variable is judged to be those with identity, the compile program and program generation method which are characterized by performing processing which generates the instruction which outputs the assumption value set as said strange variable as the result of an operation are offered.

[0027] If two or more formulas are inputted to the computer which executes the compile program concerning this invention by this, the program for making a computer compute the value of the variable concerning above-mentioned this invention will be generated.

[0028]

[Embodiment of the Invention] Hereafter, the gestalt of operation of this invention is

explained with reference to a drawing. Drawing 1 is the principle block diagram of this invention. A computer is made to perform the following processing in this invention (let the computer which performs the following processing be "variable-value decision equipment"). The following processing are started in two or more formulas for computing at least one strange variable and the default of at least one known variable contained in two or more formulas being inputted. In the example of drawing 1 , the formula "alpha=F (beta)" which asks for Variable alpha based on Variable beta, and the formula "beta=G (gamma)" which asks for Variable beta based on Variable gamma are inputted. In this example, Variable alpha and Variable beta are strange variables, and Variable gamma is a known variable. And the value "5" of Variable gamma is inputted.

[0029] First, an assumption value is set up to Variable alpha and Variable beta which are a strange variable (step S1). In the example of drawing 1 , "1" is set up as an assumption value of Variable alpha and Variable beta. A known variable, a default, and the assumption value of a strange variable are held as processing-object data 1.

[0030] Whenever an assumption value is set up to a strange variable, processing of step S2 - step S5 is performed. That is, the processing-object data 1 are copied first and the copy data 2 are generated (step S2). Next, the solution of each formula is computed based on the default of a known variable, and the assumption value set as the strange variable (step S3). The solution computed based on two or more formulas of each is newly set up as an assumption value of the strange variable computed in each formula (step S4). In the example of drawing 1 , the value of Variable alpha and Variable beta is computed and it is set as the processing-object data 1 as an assumption value. And identity with the value before and behind a setup of the assumption value of a strange variable is judged (step S5). A judgment of identity is made by comparing the contents of the processing-object data 1 and the copy data 2. For example, when the value of all variables is in agreement, it can be judged as those with identity. Moreover, when the value of the variable alpha to acquire finally is in agreement, you may make it judge it as those with identity.

[0031] In decision of identity, unless it is judged as those with identity, processing of step S2 - step S5 is performed repeatedly. In decision of identity, when the value before and behind a setup of an assumption value is judged to be those with identity, the assumption value set as the strange variable is determined as the result of an operation, and the result of an operation 3 is outputted (step S6).

[0032] If a program is created so that processing may be performed in the above procedure, regardless of the order of a publication of the formula which asks for a

variable, a right solution is computable. That is, when final solutions, such as a mathematical certification equation, are calculated, things are calculating the value of the variable which fills all formulas. Therefore, that would carry out multiple-times repeat activation of those formulas, and no value of variables will change when the formula which contains a variable from a superordinate concept to a subordinate concept is written the neither more nor less (in random order) means that the value of all the variables that fill all formulas was decided. Therefore, the value of each variable in case the value of a variable will not change can be determined as a right solution.

[0033] As mentioned above, the theory concerning this invention is simple. And a development support system is not needed, but it can apply on the level of a coding scheme, and use is easy. in addition, it is also easily possible to optimize description of the program which cared about it and indicated sequence so that the count of a repeat of calculation processing beforehand of a solution with a pre compiler (or the sequence of description at the time of activation and the sequence of activation — changing) may be lessened. Moreover, it is necessary to indicate neither IF instruction showing logic (logic), nor a CALL instruction, a GOTO instruction, etc. as a publication of a program. Therefore, the productivity of development improves wonderfully.

[0034] Here, the difference between the solution method in a current generalized computer (J. von Neumann mold) and the solution method concerning this invention is explained. When creating the program which computes the value of a certain variable, human being needs to invent the formula for computing the variable mathematically. In order to prove the formula of a superordinate concept which defines a final answer mathematically, the variable contained in the formula of a superordinate concept is defined by the subordinate concept. When it doubles with thinking of human being at this time, generally the variable and formula of a subordinate concept are gradually described from a superordinate concept. And the certification is completed in the place where the variable and formula which the subordinate concept defined reached a well-known variable and a well-known formula.

[0035] Artificer J. von Neumann of a current generalized computer (von Neumann type computer) considered as radical Motohara ** of a von Neumann type computer, carried out sequential execution of the formula containing the variable of a subordinate concept, and completed the AKI theque tea which performs the formula which finally contains the variable of a superordinate concept. This has caused the result from which the activation array of the formula in a von Neumann type computer and the order of description of the formula on thinking of human being deviate.

[0036] By the program for the conventional von Neumann type computers, the order

of a publication of a formula was doubled with thinking of human being, and the equation of a subordinate concept is gradually described from the superordinate concept. And it is made to perform sequentially from the formula of a subordinate concept by computer by using jump instruction, a call instruction, etc. abundantly. However, the bug by the mistake of the order of activation of a formula had occurred owing to the order of a publication of the formula on a program differing from the order of activation in the computer of the formula. And in the case of the large-scale software of a computer system, the amount of a program is also huge and pinpointing of the source location of a bug is difficult. Consequently, in order to discover a problem part, much time amount was spent.

[0037] On the other hand, in this invention, the programming technique which does not ask the order of activation of a formula is offered. This becomes possible to prevent generating of the bug by the mistake of the order of activation of a formula.

[0038] In addition, although it is requiring fundamentally that the value of the processing-object data 1 and the copy data 2 should be in agreement in the judgment of the identity of a solution, in the case of a floating point arithmetic, you may consider that it is the same with being approximate value. The result which depends this on a floating point arithmetic is because it becomes approximate value, even if it supplies the same value for every activation.

[0039] By the way, if the programming technique of this invention is taken, a value will be fundamentally decided sequentially from the variable of a subordinate concept. It is that a value will not change even if it computes the solution by the formula "which a value decides" here. Then, suppose hereafter that the programming technique of this invention is called a "variable sequential decision method." Moreover, suppose that the computer which performs processing in alignment with the program technique of this invention is called "the reverse von Neumann type computer (processor) of a variable sequential decision method." Furthermore, suppose that the computer which applies a variable sequential decision method among reverse J. von Neumann mold processors, and operates to juxtaposition is called "the reverse J. von Neumann mold multi-dimensional processing equipment of a variable sequential decision method."

[0040] [Gestalt of the 1st operation] drawing 2 is drawing showing the example of a hardware configuration of the reverse J. von Neumann mold processor concerning the gestalt of the 1st operation. As for the reverse J. von Neumann mold processor 100, the whole equipment is controlled by CPU101. RAM102, the hard disk drive unit (HDD) 103, the graphics processing unit 104, the input interface 105, and the communication interface 106 are connected to CPU101 through the bus 107.

[0041] A part of program of OS (Operating System) and application program [at least] which makes CPU101 perform RAM102 are stored temporarily. Moreover, various data required for processing by CPU101 are stored in RAM102. As for HDD103, OS and an application program are stored.

[0042] The monitor 11 is connected to the graphics processing unit 104. A graphics processing unit 104 displays an image on the screen of a monitor 11 according to the instruction from CPU101. The keyboard 12 and the mouse 13 are connected to the input interface 105. The input interface 105 transmits the signal sent from a keyboard 12 or a mouse 13 to CPU101 through a bus 107.

[0043] The communication interface 106 is connected to the network 14. A network 14 is a wide area network like the Internet. A communication interface 106 transmits and receives data among other computers through a network 14.

[0044] The processing facility of the gestalt of the 1st operation is realizable with the above hardware configurations. Drawing 3 is the functional block diagram of the reverse J. von Neumann mold processor concerning the gestalt of operation of this invention. With the gestalt of the 1st operation, the processing-object data storage field (X region) 21, the copy data storage field (Y region) 22, and the data storage field 23 for un-[processing] (Z region) are formed in room 20.

[0045] The X region 21 is a field for storing the variable used as a processing object. The value of two or more variables is set to the X region 21 as a continuation value. The Y region 22 is a field for storing the duplicate of the data of the X region 21. The storage capacity of the Y region 22 is the same as the X region 21.

[0046] The Z region 23 is a field for storing the data used as a processing object. The definition of the definition of input output equipment, a timer, etc. and a database are stored in the Z region 23. Moreover, the processing facility of the reverse J. von Neumann mold processor concerning the gestalt of the 1st operation divides roughly, and consists of a data input control section 110, the program execution section 120, and a data output control section 130.

[0047] The data input control section 110 performs data processing of an input system. For example, the data which answered directions by the icon of a screen, the mouse of a carbon button, etc. are set as room 20, or a data value is set up from a keyboard. Moreover, the data input control section 110 also performs a setup of data, such as a timer which changes every moment. Furthermore, the data input control section 110 answers the demand based on the actuation input from a user etc., and loads a program in room 20. For example, by the actuation input using a keyboard 12, if the activation demand which specified the file of processing in HDD103 is inputted,

the data input control section 110 acquires the specified file from HDD103, and stores it in the Z region 23 of room 20.

[0048] The program execution section 120 performs reverse J. von Neumann mold processing concerning the gestalt of the 1st operation based on the data stored in the Z region 23. With the gestalt of the 1st operation, the program execution section 120 has the output section 125 as a result of the processing-object data setting section 121, the copy section 122, the solution calculation section 123, and the coincidence judging section 124.

[0049] The processing-object data setting section 121 answers a program execution demand, out of the data stored in the Z region 23, takes out the aggregate of a variable and stores it in the X region 21. The copy section 122 copies the data of each variable of the processing object stored in the X region 21 in the Y region 22. It is after the judgment of having no identity (inequality) by the data storage back by the processing-object data setting section 121, and the coincidence judging section 124 that the copy section 122 copies data.

[0050] Out of the data stored in the Z region 23, the solution calculation section 123 acquires the formula of a variable, and computes the value of the variable of the X region 21 based on the formula. The solution calculation section 123 sets the value of the computed variable as each variable of the X region 21. Calculation processing by the solution calculation section 123 is performed after termination of the copy processing by the copy section 122.

[0051] The coincidence judging section 124 judges the identity of the data of the X region 21 and the Y region 22. The judgment of identity can be based on that the value of all variables is the same, or the value of the top variable being the same at least. Here, the top variable is a variable which is not used for a formula for the variable itself to compute other variables. Judgment processing by the coincidence judging section 124 is performed after termination of the solution calculation processing by the solution calculation section 123.

[0052] The result output section 125 copies the data of the variable stored in the X region 21 in the Z region 23. It is after the judgment with identity (coincidence) in the coincidence judging section 124 that the result output section 125 copies data.

[0053] The data output control section 130 outputs the solution of each variable stored in the Z region 23. For example, it stores in HDD103 as a file, or a document output is carried out on the screen of a display.

[0054] Next, the variable sequential decision processing in the reverse J. von Neumann mold processor of a configuration of having been shown in drawing 3 is

explained using a flow chart. Drawing 4 is a flow chart which shows the procedure of the variable sequential decision method concerning the gestalt of the 1st operation. Below, the processing shown in drawing 4 is explained along with a step number.

[0055] [Step S11] data input control section 110 loads a program to the Z region 23. At this time, the variable defined by the program is set to the X region 21.

[0056] [Step S12] copy section 122 secures the Y region 22. At this time, the copy section 122 gives initial value (it is "FF ..." in a hexadecimal) to the variable defined, and sets it as Y region. Let initial value of a variable be a different value from the variable of the X region 21.

[0057] Out of the program stored in the Z region 23, [step S13] processing-object data setting section 121 takes out the value of a variable, and sets it as each variable of the X region 21. In addition, to the variable with which the value is not set up, any value (for example, 1) is set as the Z region 23.

[0058] [Step S14] coincidence judging section 124 judges whether the value of all the variables of the Y region 22 is the same as the value of all the variables of the X region 21. If the value of the whole variable is the same, processing will be advanced to step S17. If the value of the whole variable is not the same, processing will be advanced to step S15.

[0059] [Step S15] copy section 122 copies the processing-object data of the X region 21 in the Y region 22. [Step S16] solution calculation section 123 calculates the value of the variable stored in the X region 21 based on the formula defined as the Z region 23. The value of each computed variable is set as each variable of the X region 21. Then, processing is advanced to step S14.

[0060] If judged with the variable of the X region 21, the value, and the value of the output section 125 of the variable of the Y region 22 having corresponded as a result of [step S17], it will copy in the Z region 23 by making the processing-object data of the X region 21 into a solution.

[0061] [Step S18] data output control section 130 outputs the data stored in the Z region 23 to HDD103 grade. Thus, processing of steps S14-S16 is repeated until the value of the variable of the X region 21 and the Y region 23 is in agreement. It means that the value of each variable does not change that the value of the variable of the X region 21 and the Y region 23 was in agreement, even if it substitutes and calculates a known variable at a defined ceremony. That is, it means that the value of the variable which fills the defined formula was calculated. It shall make to perform a series of processings of steps S14-S16 once into an one pass hereafter, and the count of a repeat of a series of processings shall be expressed with the count of pass.

[0062] Below, there are variables a, b, c, d, and e as an example, 105 (circle) is substituted for Variable d by 2 and Variable e, and the case where the following formulas are performed is explained.

[0063]

[Equation 4] Variable a= variable bx variable c ... (4)

[0064]

[Equation 5] Variable b= variable c+ variable d ... (5)

[0065]

[Equation 6] Variable c= variable dx variable e ... (6)

Here, a variable sequential decision method describes the program for computing the value of Variable a.

[0066] Drawing 5 is the mimetic diagram showing the 1st example of description and example of activation of a variable sequential decision method program. The contents 210 of description and the contents 220 of program execution of the program are shown among drawing.

[0067] The contents 210 of description of a program are divided into three description 211-213. The first description 211 is description for securing the field of data etc. given a definition as a Y region 22. In the example of drawing 5 , reservation of the field of Variable a, Variable b, Variable c, Variable d, and Variable e and a setup of the initial value to each variable are directed. In addition, the initial value of each variable is the value of each variable of X region, and a different value.

[0068] Description 212 is description about the definition of the data set as the X region 21. Variable a (a value is undecided), Variable b (a value is undecided), Variable c (a value is undecided), the variable d= 2, and the variable e= 105 are defined by the example of drawing 5 .

[0069] The formula showing the relation between variables is defined as description 213. Three formulas are defined by the example of drawing 5 in order of the above-mentioned equation (4), the equation (5), and the equation (6). The contents of activation of the variable sequential decision method according to the contents 210 of description of a program are shown in the contents 220 of program execution. The contents of activation are divided into three partial processings 221-223 corresponding to each of three description 211-213 of the contents 210 of description of a program.

[0070] The partial processing 221 is processing which secures the field of data etc. given a definition according to description 211. In the example of drawing 5 , the field of Variable a, Variable b, Variable c, Variable d, and Variable e is secured, and 1 is set

as each variable as initial value. Here, the initial value to set up should just be a value in which at least one variable differs from the variable of the same name of the X region 21. In the example of drawing 5 , values other than one are set as Variable d and Variable e among the variables set as the X region 21. Therefore, in the partial processing 221, the initial value of all variables is set as maximum (each cutting tool's value is FF in a hexadecimal).

[0071] The partial processing 222 is processing which defines the data to the X region 21 according to description 212. Variable a, Variable b, Variable c, Variable d, and Variable e are defined by the example of drawing 5 , and the value is set up to each variable in it. In addition, the specified value is set up to the variable as which the value is specified by description 222. To the variable as which the value is not specified by description 222, predetermined initial value (for example, 1) is set up. In the example of drawing 5 , the value of a variable a= 1, a variable b= 1, a variable c= 1, a variable d= 2, and a variable e= 105 is set up.

[0072] In addition, "0" in the formula of a computer may treat and it may become an activation error depending on the direction. That is because the operation of a formula becomes impossible when the value of the variable used as a denominator of the formula of a division is 0. Therefore, to the variable it is not decided that a value will be, it is not "0" as initial value and it is desirable to put in "1."

[0073] The partial processing 223 is processing which calculates a formula according to description 213. The contents of processing follow steps S14-S16 of drawing 4 . If it explains briefly, it is judged whether the value of all the variables of an X region =Y region is the same, and processing will be completed if the same. If not the same, the contents of the X region will be copied in Y region. And a formula is calculated in the sequence indicated by description 213. Then, it progresses to the first decision processing.

[0074] In the example of drawing 5 , the formula of a variable is defined as description 213 in order of the equation (4), the equation (5), and the equation (6). Therefore, the sequence of data processing in the partial processing 223 is also the sequence of a formula (4), a formula (5), and a formula (6).

[0075] Drawing 6 is drawing showing the transition situation of the value of the variable at the time of calculating a formula in the sequence shown in the 1st example. The initial value of five variables of the X region 21 and the Y region 22 and the value in the operation termination time of a formula are shown in drawing 6 .

[0076] In the 1st example, the continuation values of the variables a, b, c, d, and e of the X region 21 are "1, 1, 1, 2,105." All the initial value of the Y region 22 is "FFFF(s)"

(2 bytes of figure) of a hexadecimal. In addition, in drawing 6 , the decimal number shows values other than the initial value of the Y region 22.

[0077] the 1st data processing — an X region =Y region (coincidence) — it is judged with it not being in agreement by that judgment processing (step S14). Then, the contents of the X region 21 are copied in the Y region 22 (step S15), and two or more formulas which calculate the solution to are performed (step S16). As a result, Variable a serves as a value "1", Variable b serves as a value "3", and Variable c serves as a value "210." At this time, the continuation values of the variables a, b, c, d, and e of the X region 21 are "1, 3, 210, 2,105." The continuation value of the variables a, b, c, d, and e of the Y region 22 is changing from "FFFF" of a hexadecimal, and are "1, 1, 1, 2,105."

[0078] the 2nd data processing — an X region =Y region (coincidence) — it is judged with it not being in agreement by that judgment processing (step S14). Then, the contents of the X region 21 are copied in the Y region 22 (step S15), and two or more formulas which calculate the solution to are performed (step S16). As a result, Variable a serves as a value "630", Variable b serves as a value "212", and Variable c serves as a value "210." At this time, the continuation values of the variables a, b, c, d, and e of the X region 21 are "630, 212, 210, 2,105." The continuation values of the variables a, b, c, d, and e of the Y region 22 are "1, 3, 210, 2,105."

[0079] the 3rd data processing — an X region =Y region (coincidence) — in that judgment processing (step S14), it is judged with it not being in agreement. Then, the contents of the X region 21 are copied in the Y region 22 (step S15), and two or more formulas which calculate the solution to are performed (step S16). As a result, Variable a serves as a value "44520", Variable b serves as a value "212", and Variable c serves as a value "210." At this time, the continuation values of the variables a, b, c, d, and e of the X region 21 are "44520, 212, 210, 2,105." The continuation values of the variables a, b, c, d, and e of the Y region 22 are "630, 212, 210, 2,105."

[0080] the 4th data processing — an X region =Y region (coincidence) — it is judged with it not being in agreement by that judgment processing (step S14). Then, the contents of the X region 21 are copied in the Y region 22 (step S15), and two or more formulas which calculate the solution to are performed (step S16). As a result, Variable a serves as a value "44520", Variable b serves as a value "212", and Variable c serves as a value "210." At this time, the continuation values of the variables a, b, c, d, and e of the X region 21 are "44520, 212, 210, 2,105." The continuation values of the variables a, b, c, d, and e of the Y region 22 are "44520, 212, 210, 2,105." The X region 21 and the Y region 22 are in agreement in this phase.

[0081] the 5th data processing — an X region =Y region (coincidence) — by that judgment processing (step S14), it is judged with it being in agreement, the contents of the X region 21 are copied in Z region (step S17), and processing is ended. Therefore, the value at the 4th processing termination time is held as it is at the variable of the X region 21, and the variable of the Y region 22. And the value of the variable of the X region 21 is taken out as a solution.

[0082] In addition, in the example of drawing 5 and drawing 6 , the sequence that a solution cannot be found easily intentionally has described the formula. Therefore, five pass is taken to be able to find a solution. Here, processing effectiveness will be improved if sequence of a formula is replaced for example, with a pre compiler.

[0083] Next, the case where the sequence of a formula is replaced is explained as 2nd example of activation. Drawing 7 is the mimetic diagram showing the 2nd example of description and example of activation of a variable sequential decision method program. The contents 230 of description and the contents 240 of program execution of the program are shown among drawing. The contents 230 of description of drawing 7 consist of three description 231–233. Description 231,232 is the same as the description 211,212 shown in drawing 5 . As for description 233, the formula is defined in order of the equation (5), the equation (4), and the equation (6).

[0084] Three partial processings 241–243 are consisted of by the contents 240 of program execution. The partial processing 241,242 is the same processing as the partial processings 221 and 222 shown in drawing 5 . The partial processing 243 is processing which calculates a formula according to description 233. Although the contents of processing of the partial processing 243 are the partial processing 223 shown in drawing 5 , and the almost same processing, only the sequence of data processing differs. In the partial processing 243, an operation is performed in order of a formula (5), a formula (4), and a formula (6).

[0085] Drawing 8 is drawing showing the transition situation of the value of the variable at the time of calculating a formula in the sequence shown in the 2nd example. The initial value of five variables of the X region 21 and the Y region 22 and the value in the operation termination time of a formula are shown in drawing 8 .

[0086] In the 2nd example, the initial value of the variables a, b, c, d, and e of the X region 21 is "1, 1, 1, 2,105." All the initial value of the Y region 22 is "FFFF(s)" (2 bytes of figure) of a hexadecimal. In addition, in drawing 8 , the decimal number shows values other than the initial value of the Y region 22.

[0087] the 1st data processing — an X region =Y region (coincidence) — it is judged with it not being in agreement by that judgment processing (step S14). Then, the

contents of the X region 21 are copied in the Y region 22 (step S15), and two or more formulas which calculate the solution to are performed (step S16). As a result, Variable a serves as a value "3", Variable b serves as a value "3", and Variable c serves as a value "210." At this time, the continuation values of the variables a, b, c, d, and e of the X region 21 are "3, 3, 210, 2,105." The continuation value of the variables a, b, c, d, and e of the Y region 22 is changing from "FFFF" of a hexadecimal, and are "1, 1, 1, 2,105."

[0088] the 2nd data processing — an X region =Y region (coincidence) — it is judged with it not being in agreement by that judgment processing (step S14). Then, the contents of the X region 21 are copied in the Y region 22 (step S15), and two or more formulas which calculate the solution to are performed (step S16). As a result, Variable a serves as a value "44520", Variable b serves as a value "212", and Variable c serves as a value "210." At this time, the continuation values of the variables a, b, c, d, and e of the X region 21 are "44520, 212, 210, 2,105." The continuation values of the variables a, b, c, d, and e of the Y region 22 are "3, 3, 210, 2,105."

[0089] the 3rd data processing — an X region =Y region (coincidence) — in that judgment processing (step S14), it is judged with it not being in agreement. Then, the contents of the X region 21 are copied in the Y region 22 (step S15), and two or more formulas which calculate the solution to are performed (step S16). As a result, Variable a serves as a value "44520", Variable b serves as a value "212", and Variable c serves as a value "210." At this time, the continuation values of the variables a, b, c, d, and e of the X region 21 are "44520, 212, 210, 2,105." The continuation values of the variables a, b, c, d, and e of the Y region 22 are "44520, 212, 210, 2,105." The X region 21 and the Y region 22 are in agreement in this phase.

[0090] the 4th data processing — an X region =Y region (coincidence) — by that judgment processing (step S14), it is judged with it being in agreement, the contents of the X region 21 are copied in Z region (step S17), and processing is ended. Therefore, the value at the 3rd processing termination time is held as it is at the variable of the X region 21, and the variable of the Y region 22. And the value of the variable of the X region 21 is taken out as a solution.

[0091] Thus, a right solution can be acquired even if it replaces the order of a publication of a formula (4) and a formula (5). Moreover, from the formula (5), since the direction of a formula (4) is the formula of a superordinate concept, the count of an operation is decreasing by having indicated the formula (5) previously.

[0092] Next, the example at the time of indicating in an order from the formula of a subordinate concept is explained as the 3rd example. Drawing 9 is the mimetic

diagram showing the 3rd example of description and example of activation of a variable sequential decision method program. The contents 250 of description and the contents 260 of program execution of the program are shown among drawing. The contents 250 of description of drawing 9 consist of three description 251-253. Description 251,252 is the same as the description 211,212 shown in drawing 5 . As for description 253, the formula is defined in order of the equation (6), the equation (5), and the equation (4).

[0093] The contents 260 of program execution consist of three partial processings 261-263. The partial processing 261,262 is the same processing as the partial processings 221 and 222 shown in drawing 5 . The partial processing 263 is processing which calculates a formula according to description 253. Although the contents of processing of the partial processing 263 are the partial processing 223 shown in drawing 5 , and the almost same processing, only the sequence of data processing differs. In the partial processing 263, an operation is performed in order of a formula (6), a formula (5), and a formula (4).

[0094] Drawing 10 is drawing showing the transition situation of the value of the variable at the time of calculating a formula in the sequence shown in the 3rd example. The initial value of five variables of the X region 21 and the Y region 22 and the value in the operation termination time of a formula are shown in drawing 10 .

[0095] In the 3rd example, the initial value of the variables a, b, c, d, and e of the X region 21 is "1, 1, 1, 2,105." All the initial value of the Y region 22 is "FFFF(s)" (2 bytes of figure) of a hexadecimal. In addition, in drawing 10 , the decimal number shows values other than the initial value of the Y region 22.

[0096] the 1st data processing — an X region =Y region (coincidence) — it is judged with it not being in agreement by that judgment processing (step S14). Then, the contents of the X region 21 are copied in the Y region 22 (step S15), and two or more formulas which calculate the solution to are performed (step S16). As a result, Variable a serves as a value "44520", Variable b serves as a value "212", and Variable c serves as a value "210." At this time, the continuation values of the variables a, b, c, d, and e of the X region 21 are "44520, 212, 210, 2,105." The continuation value of the variables a, b, c, d, and e of the Y region 22 is changing from "FFFF" of a hexadecimal, and are "1, 1, 1, 2,105."

[0097] the 2nd data processing — an X region =Y region (coincidence) — in that judgment processing (step S14), it is judged with it not being in agreement. Then, the contents of the X region 21 are copied in the Y region 22 (step S15), and two or more formulas which calculate the solution to are performed (step S16). As a result,

Variable a serves as a value "44520", Variable b serves as a value "212", and Variable c serves as a value "210." At this time, the continuation values of the variables a, b, c, d, and e of the X region 21 are "44520, 212, 210, 2,105." The continuation values of the variables a, b, c, d, and e of the Y region 22 are "44520, 212, 210, 2,105." The X region 21 and the Y region 22 are in agreement in this phase.

[0098] the 3rd data processing — an X region =Y region (coincidence) — by that judgment processing (step S14), it is judged with it being in agreement, the contents of the X region 21 are copied in Z region (step S17), and processing is ended. Therefore, the value at the 2nd processing termination time is held as it is at the variable of the X region 21, and the variable of the Y region 22. And the value of the variable of the X region 21 is taken out as a solution.

[0099] Thus, the count of an operation can be decreased by describing in an order from the formula of a subordinate concept most. By the way, description of operation instruction is in random order in a variable sequential decision method. Therefore, the sequence it is easier for human being to understand can describe a formula. And it is the range which human being thinks of, and it is also possible to change the order of a publication of a formula and to aim at improvement in the engine performance. Namely, what is necessary is just to determine sequence suitably by the best technique which an operator thinks of, since there is no constraint about exchange of the sequence of a formula.

[0100] For example, the engine performance is improvable by determining the order of description of a formula as order with few variables contained in a formula. That is, when it expresses the formula of complicated structure with a layered structure, low order structure becomes a simple equation in many cases. Then, if it assumes that it is such a simple equation that there are few variables and the formula is described in order with few variables, in many cases, the increase in efficiency of processing can be attained.

[0101] As explained above, in the reverse J. von Neumann mold processor which applied the variable sequential decision method, two or more variables are expressed with arithmetic expression, such as a four rules operator, as a publication of a program, and a right solution is acquired irrespective of the order of activation of the arithmetic expression. That is, a programmer does not need to define the order of activation. Generally, in order to define the order of program execution, the instruction showing the logic (logic) of IF instruction for coming and going the line (location) the program is described to be, a CALL instruction, a GOTO instruction, etc. is used. However, according to the variable sequential decision method concerning the gestalt of this

operation, the publication of the instruction which defines the order of activation becomes unnecessary. Thereby, generating of the bug by the definition mistake of the order of activation is lost, and simplification of a debugging activity is attained. Consequently, productivity improves wonderfully.

[0102] Furthermore, that the definition of the order of activation is unnecessary contributes also to few quantification of the amount of description of a program (number of steps). If the gestalt of the 1st operation is used, the amount of description of a program will drop to 1/3 - 1/5 compared with the former. Thereby, an activity required for production of software drops to several [1/].

[0103] Moreover, since it completes by carrying out the test process (systems testing) corresponding to a high order process (systems design) by losing complicated logic like a programming process, a test process decreases sharply.

[0104] In addition, the variable sequential decision method concerning the gestalt of the 1st operation is applicable to debugging of a conventional-type program. That is, it is described by the program as usual that it generally performs previously from the formula which computes the variable of a subordinate concept. And the formula which computes the variable of a superordinate concept is performed at the end. Therefore, if the sequence of a formula is right when a program is conventionally performed in processing of step S16 shown in drawing 4 , each variable will carry out sequential decision by the two pass eye, and processing termination will be carried out by 3 pass eye. That is, it does not end with three pass, but if there is a program conventionally which performs 4 pass eye or subsequent ones, it can be concluded that the cause of a bug lurks in somewhere.

[0105] It is because the value needed for all variables by the one-pass eye is decided by the conventional-type program programmed correctly so that it may perform sequentially from the formula of a subordinate concept, the processing-object data of the X region 21 and the copy data of the Y region 22 are in agreement by the two pass eye and the judgment of coincidence should be performed by 3 pass eye.

[0106] Debugging of the conventional-type program by such variable sequential decision method concerning the gestalt of the 1st operation is good to use at a combined test (test which unifies two or more programs and performs activation check) - maintenance process (activity which bases and applies to business and corrects a trouble). If it furthermore says and the fair copy of branch instruction unclear for human beings, such as a CALL instruction which branches in the distance, will be removed and made in a conventional-type program, it is changeable into an intelligible program.

[0107] moreover, about the real number operation used by the program for scientific calculation "X region = in order to make it cease by that judgment processing" (step S14) Y regions (coincidence), it may have that the variable a of the last purpose of the X region 21 and the variable a of the last purpose of the Y region 22 became the same value (end value), and you may judge it as that with which the "X region =Y region" was filled. That is, processing until the whole variable county is in agreement is used at the times, such as an integer. This is processing for the property (precision is sacrificed by improvement in the engine performance) of a real type floating point unit. In addition, also in the variable of not only a real type operation but integer type, it may have that the variable a of the last purpose of the X region 21 and the variable a of the last purpose of the Y region 22 became the same value (end value), and you may judge it as that with which the "X region =Y region" was filled. In this case, processing effectiveness improves.

[0108] [The gestalt of the 2nd operation], next the gestalt of operation of the 2nd of this invention are explained. The gestalt of the 2nd operation is an operation gestalt in the case of performing processing concerning a variable sequential decision method to juxtaposition. With the gestalt of the 2nd operation, multiplexing of processing is attained by applying processing of a variable sequential conversion method to the multi-dimensional processing equipment indicated by JP,56-97170,A. Below, in the gestalt of the 2nd operation, a different part from the gestalt of the 1st operation is explained.

[0109] Drawing 11 is the block diagram showing the example of a hardware configuration of the reverse J. von Neumann mold multi-dimensional processing equipment concerning the gestalt of the 2nd operation. In drawing 11 , the same sign is given to the same component as the hardware configuration of the gestalt of the 1st operation shown in drawing 2 , and explanation is omitted.

[0110] Two or more CPU 101a-101c is formed in reverse J. von Neumann mold multi-dimensional processing equipment 100a in the gestalt of the 2nd operation. These CPUs 101a-101c can perform processing about a variable sequential conversion method to juxtaposition. Components other than CPU101a - 101c of reverse J. von Neumann mold multi-dimensional processing equipment 100a have the same function as the component of a same name shown in drawing 2 .

[0111] Thus, the parallel processing of a variable sequential conversion method becomes possible using the hardware of a multi-CPU. With the gestalt of the 2nd operation, processing to each CPU shall be portioned out using a decision table.

[0112] Drawing 12 is the functional block diagram of the gestalt of the 2nd operation.

In drawing 12 , the same sign is given to the same component as functional block of the gestalt of the 1st operation shown in drawing 3 , and explanation is omitted.

[0113] With the reverse J. von Neumann mold multi-dimensional processing equipment of the gestalt of the 2nd operation, it has the data input control section 110, program execution section 120a, and the data output control section 130. In addition, the decision table 140,150 is contained in the program which the data input control section 110 loads to room 20.

[0114] Two or more decision tables 140,150 are stored in room 20. As for the decision table 140,150, conditional statement (condition entry), its reply (condition stub), and activation directions (action stub) of a run command (action entries) and its instruction are defined. The detail of a decision table 140,150 is mentioned later.

[0115] Program execution section 120a consists of output section 125a as a result of processing-object data setting section 121a, copy section 122a, two or more solution calculation sections 123a-123c, and coincidence judging section 124a. Each of these components have the same function as the component of the same name of the gestalt of the 1st operation shown in drawing 3 . However, each component shown in drawing 12 judges whether processing is performed or not with reference to a decision table 140,150.

[0116] Moreover, the solution calculation sections 123a-123c are the processing facilities prepared in every CPU101a - 101c. That is, each CPU has the function of the solution calculation section, and can perform processing to juxtaposition by two or more CPU 101a-101c.

[0117] Drawing 13 is drawing showing an example of a decision table. As shown in drawing 13 , the decision table 140 in the gestalt of the 2nd operation is a decision table of pretreatment and after treatment, and a decision table 150 is a decision table of a variable sequential decision method.

[0118] The conditional statement "whether it is processing initiation", and the conditional statement "whether it is processing termination" are registered into the condition entry 141 of a decision table 140. In the condition stub 142 of a decision table 140, "YES" is set as the reply column (the 1st train) of the left-hand side matched with the conditional statement "whether it is processing initiation", and the reply is not set to the right-hand side reply column (the 2nd train). Moreover, a reply is not set to the reply column (the 1st train) of the left-hand side matched with the conditional statement "whether it is processing termination", but "YES" is set to the right-hand side reply column (the 2nd train).

[0119] The run command of "a Z region ->X region (the contents of the Z region are

copied)" and "a post process (X region -> Z region copy)" is registered into the action entries 143 of a decision table 140. In the action stub 144 of a decision table 140, "1" is set as the activation directions column (the 1st train) of the left-hand side matched with the run command "a Z region ->X region (the contents of the Z region are copied)", and activation directions are not set to right-hand side activation directions (the 2nd train). Moreover, in the action stub 144, activation directions are not set to the activation directions column (the 1st train) of the left-hand side matched with the conditional statement "a post process (X region -> Z region copy)", but "1" is set to the right-hand side activation directions column (the 2nd train).

[0120] The conditional statement of "an X region =Y region (was it in agreement?)" is registered into the condition entry 151 of a decision table 150. It matches with the conditional statement of "an X region =Y region (was it in agreement?)", and two reply columns are prepared in the condition stub 152 of a decision table 150. "YES" is set to the left-hand side reply column (the 1st train). "NO" is set to the right-hand side reply column (the 2nd train).

[0121] The run command of "an X region ->Y region (the contents of the X region are copied)", the "variable a= variable bx variable c", the "variable b= variable c+ variable d", the "variable c= variable dx variable e", the "variable f= variable gx variable h", and "EXIT processing" is registered into the action entries 153 of a decision table 150. In the action stub 154 of a decision table 150, activation directions are not set to the activation directions column (the 1st train) of the left-hand side matched with the run command of "an X region ->Y region (the contents of the X region are copied)", but "1" is set to right-hand side activation directions (the 2nd train). Activation directions are not set to the activation directions column (the 1st train) of the left-hand side matched with the run command of "the variable a= variable bx variable c", but "2" is set to right-hand side activation directions (the 2nd train). Activation directions are not set to the activation directions column (the 1st train) of the left-hand side matched with the run command of "the variable b= variable c+ variable d", but "3" is set to right-hand side activation directions (the 2nd train). Activation directions are not set to the activation directions column (the 1st train) of the left-hand side matched with the run command of "the variable c= variable dx variable e", but "4" is set to right-hand side activation directions (the 2nd train). Activation directions are not set to the activation directions column (the 1st train) of the left-hand side matched with the run command of "the variable f= variable gx variable h", but "2" is set to right-hand side activation directions (the 2nd train). And "1" is set to the activation directions column (the 1st train) of the left-hand side matched with the run

command of "EXIT processing", and activation directions are not set to right-hand side activation directions (the 2nd train).

[0122] In the example of drawing 13 , "YES" or "NO" is set to the condition stub. If a condition stub is "YES", the action entries by which the numeric value is set as the action stub of the train as a truth (conditions were fulfilled) case with the same processing result of the conditional statement of a corresponding condition entry will be performed. If a condition stub is "NO", the action entries by which the numeric value is set as the action stub of the train with the processing result of the conditional statement of a corresponding condition entry same in a fake (conditions are not fulfilled) case will be performed.

[0123] The numeric value is set to the action stub. Each numeric value expresses the order of activation. The numeric values of the same train of an action stub are compared, and the action entries corresponding to numerical small order are performed. Here, since there are no superiority or inferiority in processing of those action entries when the numeric value of an action stub is the same, it can perform by being parallel in a different CPU. In the example of drawing 13 , the run command of "the variable a= variable bx variable c" and the run command of "the variable f= variable gx variable h" are parallel, and it can process.

[0124] The following processings are performed in the reverse J. von Neumann mold multi-dimensional processing equipment of the above configurations. First, the data input control section 110 loads a program to room 20. A decision table 140,150 is contained in the loaded program. Program execution section 120a performs processing based on the program loaded to room 20. That is, processing when processing-object data setting section 121a which has recognized that loading of a program was completed fills the conditional statement of "whether to be processing initiation" with reference to a decision table 140 (condition stub "YES") is judged. In the decision table 140 shown in drawing 13 , since "YES" is set as the 1st train of the condition stub 142, the 1st train of an action stub 144 is referred to. Then, processing of "a Z region ->X region (the contents of the Z region are copied)" in which 1 is set as the 1st train of an action stub 144 is performed by processing-object data setting section 121a. That is, processing-object data setting section 121a sets the value of each variable set as Z region 23a as the variable of the X region 21. Then, copy section 122a secures the Y region 22.

[0125] Next, decision processing of the conditional statement "an X region =Y region (was it in agreement?)" set as the condition entry 151 of a decision table 150 is performed by coincidence judging section 124a.

[0126] If the decision result of "an X region =Y region (was it in agreement?)" is "NO" (the 2nd train of the condition stub 152), a decision table 150 will be referred to and processing which met the run command in the action entries 153 by which the figure is set as the 2nd train of an action stub 154 will be performed. Processing is performed by the small order of the figure set as the action stub 154. Moreover, when the figure set as the action stub 154 is the same, juxtaposition executes those instructions by two or more solution calculation sections 123a-123c.

[0127] If the decision result of "an X region =Y region (was it in agreement?)" is "YES" (the 1st train of the condition stub 152), a decision table 150 will be referred to and EXIT processing to which "1" is set as the 1st train of an action stub 154 will be performed. EXIT processing is processing which notifies that the solution was decided from coincidence judging section 124a to output section 125a a result.

[0128] Then, result output section 125a judges the conditional statement of "whether to be processing termination" to be "YES" (the 2nd train of the condition stub 142) with reference to the condition entry 141 of a decision table 140. and the instruction "a post process (X region -> Z region copy)" with which "1" is set as the 2nd train of an action stub 144 — a result — output section 125a — performing . This processing is processing which copies as a solution the value of each variable set as the X region 21 to Z region 23a.

[0129] Then, the data output control section 130 outputs the solution stored in Z region 23a. Thus, processing of a variable sequential decision method can be processed to juxtaposition by two or more CPUs by using a decision table. That is, it is possible to entrust the processing which computes the solution to, therefore a parallel processing machine, and to measure the improvement in the engine performance.

[0130] In addition, not only a formula but the run command described by the action entries of a decision table can describe the address of the Internet or intranet. And the infrastructure of the present router equipment or a LAN cable can be made into a super parallel machine by the ability to be burned as LSI. In an operating system, if a browser, Java (trademark) language, etc. occur, performing multi-dimensional processing will be realized easily.

[0131] Although the programme description by the variable sequential decision method and the contents of activation of the computer according to the description were explained, a computer can also be made to generate automatically the programme description by the variable sequential decision method by translation (compile) in the example of the [gestalt of the 3rd operation] above. Suppose that such a computer is called program generation equipment. Below, it explains as a

gestalt of the 3rd operation of generation processing of the program by the variable sequential decision method.

[0132] Drawing 14 is the mimetic diagram of generation processing of a variable sequential decision program. If the formula group 310 is inputted as shown in drawing, program generation processing (S301) will be performed in program generation equipment.

[0133] In program generation processing, the instruction executed is generated in response to the input of the value (default) of at least one variable (known variable) contained in two or more formulas. First, the instruction (processing-object data setting instruction) 321 which sets processing-object data (value of two or more variables) as each variable of X region is generated. At this time, the instruction for setting up a temporary value (assumption value) is generated to the variable (strange variable) with which the value is not set up. A temporary value is "1." Next, the instruction (copy statement) 322 which copies the data of X region in Y region is generated. Next, whenever an assumption value is set up to a strange variable, the instruction (solution calculation instruction) 323 which makes the solution of each formula compute based on the default of a known variable and the assumption value set as the strange variable is generated. Next, the instruction (assumption value setting instruction) 324 to which the solution computed based on two or more formulas of each is made to newly set as an assumption value of the predetermined variable computed in each formula is generated. Next, whenever an assumption value is set up to a strange variable, the instruction (coincidence judging instruction) 325 which makes identity with the value before and behind a setup of the assumption value of a strange variable judge is generated. Finally, when the value after a setup of an assumption value is judged to be the same, the instruction (result output instruction) 326 which determines the assumption value set as the strange variable as the result of an operation is generated.

[0134] Each generated instruction is arranged in order of generation, and serves as the variable sequential decision program 320. The variable sequential decision program 320 is performed according to the input of a variable value 330 (step S302). The contents of activation at this time are as having been shown in the flow chart of drawing 4 . By performing the variable sequential decision program 320, the solution 340 according to the inputted variable value 330 is generated.

[0135] Thus, the variable sequential decision program 320 is generable only by inputting two or more formulas. That is, it can compile. In addition, the variable sequential decision program 320 can also be generated in an absolute language, and

can also be generated with programming language, such as C and BASIC. When generating the variable sequential decision program 320 with programming language, such as C and BASIC, program generation processing (step S301) will perform pre compile. Thereby, it becomes easy to apply a variable sequential decision method to a part of program described by C etc.

[0136] With the variable sequential decision method it is indicated to the gestalt of operation of this invention that explained beyond [the effectiveness of the gestalt of operation], the agreement of the sequence of complicated processing is made unnecessary, and the software which a bug does not have simply for a short period of time, either can be developed, and can be worked. Thereby, improvement in the quality of software, the fall of software-development cost, and compaction of a development cycle are attained. That is, the agreement mistake of the order of activation of a formula is lost, and it becomes discovery and correctable easily by the automatic test of a data injection method about the entry mistake of the four rules operator which are other mistakes, and the entry mistake of a variable name.

[0137] In addition, if processing of a variable sequential decision method is applied with the description approach of a conventional-type program, as long as the order of activation of a formula is a right array, each variable carries out sequential decision by the two pass, and it becomes processing termination by 3 pass eye. That is, if there is a program which performs 4 pass eye or subsequent ones, and does not cease, it can be concluded that the cause of a bug lurks in somewhere.

[0138] Moreover, it becomes possible to measure the improvement in the engine performance by performing the program of a variable sequential decision method with the multi-dimensional processing equipment which applied the decision table to parallel processing. Moreover, with the gestalt of the above-mentioned implementation, the value of two or more variables is set to X region and Y region as a continuation value. Thereby, comparison processing can be performed at a high speed.

[0139] The above-mentioned processing facility which is [offer of the program which realizes a processing facility] is realizable by computer. In that case, the program which described the contents of processing of the function which a reverse J. von Neumann mold processor, reverse J. von Neumann mold multi-dimensional processing equipment, and program generation equipment should have is offered. By executing the program by computer, the above-mentioned processing facility is realized on a computer. The program which described the contents of processing is recordable on the record medium which can be read by computer. As a record medium which can be read, there are a magnetic recording medium, an optical disk, a

magneto-optic-recording medium, semiconductor memory, etc. by computer. There are a hard disk drive unit (HDD), a flexible disk (FD), a magnetic tape, etc. in a magnetic recording medium. There is CD-R [DVD (Digital Versatile Disc), DVD-RAM (Random Access Memory), CD-ROM (Compact Disc Read Only Memory), and] (Recordable)/RW (ReWritable) etc. in an optical disk. There is MO (Magneto-Optical disk) etc. in a magneto-optic-recording medium.

[0140] When circulating a program, portable mold record media with which the program was recorded, such as DVD and CD-ROM, are sold, for example. Moreover, the program is stored in the store of a server computer and the program can also be transmitted to other computers from a server computer through a network.

[0141] The computer which executes a program stores in self storage the program transmitted from the program or server computer recorded for example, on the portable mold record medium. And a computer reads a program in self storage and performs processing according to a program. In addition, a computer can read a direct program in a portable mold record medium, and can also perform processing according to the program. Moreover, a computer can also perform processing which followed the received program serially, whenever a program is transmitted from a server computer.

[0142] (Additional remark 1) In the program for making a computer compute the value of a variable Two or more formulas for computing at least one strange variable to said computer, If the default of at least one known variable contained in said two or more formulas is inputted Whenever it sets up an assumption value to said strange variable and an assumption value is set up to said strange variable It is based on the default of said known variable, and the assumption value set as said strange variable. Said solution which repeated and computed the solution of each of said formula and was computed based on said each of two or more formulas It newly sets up as an assumption value of said strange variable computed in said each formula. Whenever an assumption value is set up to said strange variable, the identity of the value before and behind a setup of the assumption value of said strange variable is judged. The program characterized by performing processing which outputs the assumption value set as said strange variable as the result of an operation when the value before and behind a setup of the assumption value of said strange variable is judged to be those with identity.

[0143] (Additional remark 2) Program of the additional remark 1 publication characterized by judging it as those with identity in decision of said identity when the value before and behind a setup of the assumption value of said all strange variables is in agreement.

[0144] (Additional remark 3) Program of the additional remark 1 publication characterized by judging it as those with identity in decision of said identity when the value before and behind a setup of the assumption value of some strange variables in two or more strange variables is in agreement.

[0145] (Additional remark 4) Program of the additional remark 1 publication characterized by judging it as those with identity in decision of said identity when the value before and behind a setup of the assumption value of said strange variable turns into approximate value.

[0146] (Additional remark 5) Program of the additional remark 1 publication characterized by judging the identity of the value before and behind a setup of the assumption value of said strange variable by holding the copy of the assumption value of said strange variable before a setup, and comparing said copy with the assumption value set as said strange variable by decision of said identity before an assumption value is set up to said strange variable.

[0147] (Additional remark 6) Program of the additional remark 1 publication characterized by being parallel and computing the solution to in calculation of the solution of each of said formula using two or more processors.

(Additional remark 7) Program of the additional remark 6 publication characterized by making juxtaposition perform calculation of the solution of two or more formulas which do not have superiority or inferiority of the order of processing in the decision table with which the sequence of the processing which should be performed was specified in calculation of the solution of each of said formula by mutually different processor.

[0148] (Additional remark 8) In the compile program for generating the program the instruction for making the value of a variable compute was described to be The computer into which two or more formulas for computing at least one strange variable were inputted is answered at the input of the default of at least one known variable contained in said two or more formulas. Whenever it generates the instruction which sets up an assumption value to said strange variable and an assumption value is set up to said strange variable It is based on the default of said known variable, and the assumption value set as said strange variable. Said solution which generated the instruction which repeats and computes the solution of each of said formula, and was computed based on said each of two or more formulas The instruction newly set up as an assumption value of said strange variable computed in said each formula is generated. Whenever an assumption value is set up to said strange variable, the instruction which judges the identity of the value before and behind a setup of the assumption value of said strange variable is generated. The compile program

characterized by performing processing which generates the instruction which outputs the assumption value set as said strange variable as the result of an operation when the value before and behind a setup of the assumption value of said strange variable is judged to be those with identity.

[0149] (Additional remark 9) Each instruction generated is the compile program of the additional remark 8 publication characterized by what is described by the source code of predetermined programming language.

(Additional remark 10) Each instruction generated is the compile program of the additional remark 8 publication characterized by what is described in the absolute language.

[0150] (Additional remark 11) Two or more formulas for computing at least one strange variable in the variable-value decision approach by the computer, If the default of at least one known variable contained in said two or more formulas is inputted Whenever it sets up an assumption value to said strange variable and an assumption value is set up to said strange variable It is based on the default of said known variable, and the assumption value set as said strange variable. Said solution which repeated and computed the solution of each of said formula and was computed based on said each of two or more formulas It newly sets up as an assumption value of said strange variable computed in said each formula. Whenever an assumption value is set up to said strange variable, the identity of the value before and behind a setup of the assumption value of said strange variable is judged. The variable-value decision approach characterized by what the assumption value set as said strange variable is outputted for as the result of an operation when the value before and behind a setup of the assumption value of said strange variable is judged to be those with identity.

[0151] (Additional remark 12) In the program generation method for generating the program for making the value of said strange variable compute based on two or more formulas for computing at least one strange variable The input of the default of at least one known variable contained in said two or more formulas is answered. Whenever it generates the instruction which sets up an assumption value to said strange variable and an assumption value is set up to said strange variable It is based on the default of said known variable, and the assumption value set as said strange variable. Said solution which generated the instruction which repeats and computes the solution of each of said formula, and was computed based on said each of two or more formulas The instruction newly set up as an assumption value of said strange variable computed in said each formula is generated. Whenever an assumption value is set up to said strange variable, the instruction which judges the identity of the value

before and behind a setup of the assumption value of said strange variable is generated. The program generation method characterized by what the instruction which outputs the assumption value set as said strange variable as the result of an operation is generated for when the value before and behind a setup of the assumption value of said strange variable is judged to be those with identity.

[0152] (Additional remark 13) Two or more formulas for computing at least one strange variable in the variable-value decision equipment which computes the value of a variable, If the default of at least one known variable contained in said two or more formulas is inputted Whenever an assumption value is set up to the 1st setting means which sets up an assumption value to said strange variable, and said strange variable A calculation means to repeat and compute the solution of each of said formula based on the default of said known variable, and the assumption value set as said strange variable, The 2nd setting means which newly sets up said solution computed based on said each of two or more formulas as an assumption value of said strange variable computed in said each formula, A decision means to judge the identity of the value before and behind a setup of the assumption value of said strange variable whenever an assumption value is set up to said strange variable, Variable-value decision equipment characterized by having an output means to output the assumption value set as said strange variable as the result of an operation when the value before and behind a setup of the assumption value of said strange variable is judged to be those with identity.

[0153] (Additional remark 14) In the program generation equipment for generating the program for making the value of said strange variable compute based on two or more formulas for computing at least one strange variable The input of the default of at least one known variable contained in said two or more formulas is answered. The 1st instruction generation means which generates the instruction which sets up an assumption value to said strange variable, The 2nd instruction generation means which generates the instruction which repeats and computes the solution of each of said formula based on the default of said known variable, and the assumption value set as said strange variable whenever an assumption value is set up to said strange variable, The 3rd instruction generation means which generates the instruction which newly sets up said solution computed based on said each of two or more formulas as an assumption value of said strange variable computed in said each formula, The 4th instruction generation means which generates the instruction which judges the identity of the value before and behind a setup of the assumption value of said strange variable whenever an assumption value is set up to said strange variable, Program

generation equipment characterized by having the 5th instruction generation means which generates the instruction which outputs the assumption value set as said strange variable as the result of an operation when the value before and behind a setup of the assumption value of said strange variable is judged to be those with identity.

[0154] (Additional remark 15) In the record medium which recorded the program for making the value of a variable compute and in which computer reading is possible Two or more formulas for computing at least one strange variable to said computer, If the default of at least one known variable contained in said two or more formulas is inputted Whenever it sets up an assumption value to said strange variable and an assumption value is set up to said strange variable It is based on the default of said known variable, and the assumption value set as said strange variable: Said solution which repeated and computed the solution of each of said formula and was computed based on said each of two or more formulas It newly sets up as an assumption value of said strange variable computed in said each formula. Whenever an assumption value is set up to said strange variable, the identity of the value before and behind a setup of the assumption value of said strange variable is judged. The record medium characterized by performing processing which outputs the assumption value set as said strange variable as the result of an operation when the value before and behind a setup of the assumption value of said strange variable is judged to be those with identity.

[0155] (Additional remark 16) In the record medium which recorded the compile program for generating the program the instruction for making the value of a variable compute was described to be and in which computer reading is possible Said computer into which two or more formulas for computing at least one strange variable were inputted is answered at the input of the default of at least one known variable contained in said two or more formulas. Whenever it generates the instruction which sets up an assumption value to said strange variable and an assumption value is set up to said strange variable It is based on the default of said known variable, and the assumption value set as said strange variable. Said solution which generated the instruction which repeats and computes the solution of each of said formula, and was computed based on said each of two or more formulas The instruction newly set up as an assumption value of said strange variable computed in said each formula is generated. Whenever an assumption value is set up to said strange variable, the instruction which judges the identity of the value before and behind a setup of the assumption value of said strange variable is generated. The record medium

characterized by performing processing which generates the instruction which outputs the assumption value set as said strange variable as the result of an operation when the value before and behind a setup of the assumption value of said strange variable is judged to be those with identity.

[0156]

[Effect of the Invention]

When the solution of two or more formulas was repeated and computed with the default of a fixed variable, and the assumption value of an undecided variable in this invention as explained above, and identity was in the value of the assumption value in calculation order, it was made to output the assumption value then set as the strange variable as the result of an operation. Therefore, a right solution can be acquired regardless of the order of count of two or more formulas. Thereby, generating of the bug by the mistake of the order of count can be prevented. And optimization can be easily attained by rearranging the order of count of a formula by the suitable approach.

DESCRIPTION OF DRAWINGS

[Brief Description of the Drawings]

[Drawing 1] It is the principle block diagram of this invention.

[Drawing 2] It is the drawing showing the example of a hardware configuration of the reverse J. von Neumann mold processor concerning the gestalt of the 1st operation.

[Drawing 3] It is the functional block diagram of the reverse J. von Neumann mold processor concerning the gestalt of operation of this invention.

[Drawing 4] It is the flow chart which shows the procedure of the variable sequential decision method concerning the gestalt of the 1st operation.

[Drawing 5] It is the mimetic diagram showing the 1st example of description and example of activation of a variable sequential decision method program:

[Drawing 6] It is drawing showing the transition situation of the value of the variable at the time of calculating a formula in the sequence shown in the 1st example.

[Drawing 7] It is the mimetic diagram showing the 2nd example of description and example of activation of a variable sequential decision method program.

[Drawing 8] It is drawing showing the transition situation of the value of the variable at the time of calculating a formula in the sequence shown in the 2nd example.

[Drawing 9] It is the mimetic diagram showing the 3rd example of description and example of activation of a variable sequential decision method program.

[Drawing 10] It is drawing showing the transition situation of the value of the variable at the time of calculating a formula in the sequence shown in the 3rd example.

[Drawing 11] It is the block diagram showing the example of a hardware configuration of the reverse J. von Neumann mold multi-dimensional processing equipment concerning the gestalt of the 2nd operation.

[Drawing 12] It is the functional block diagram of the gestalt of the 2nd operation.

[Drawing 13] It is drawing showing an example of a decision table.

[Drawing 14] It is the mimetic diagram of generation processing of a variable sequential decision program.

[Drawing 15] It is drawing showing the example of description and the example of activation of a conventional-type program.

[Description of Notations]

1 Processing-Object Data

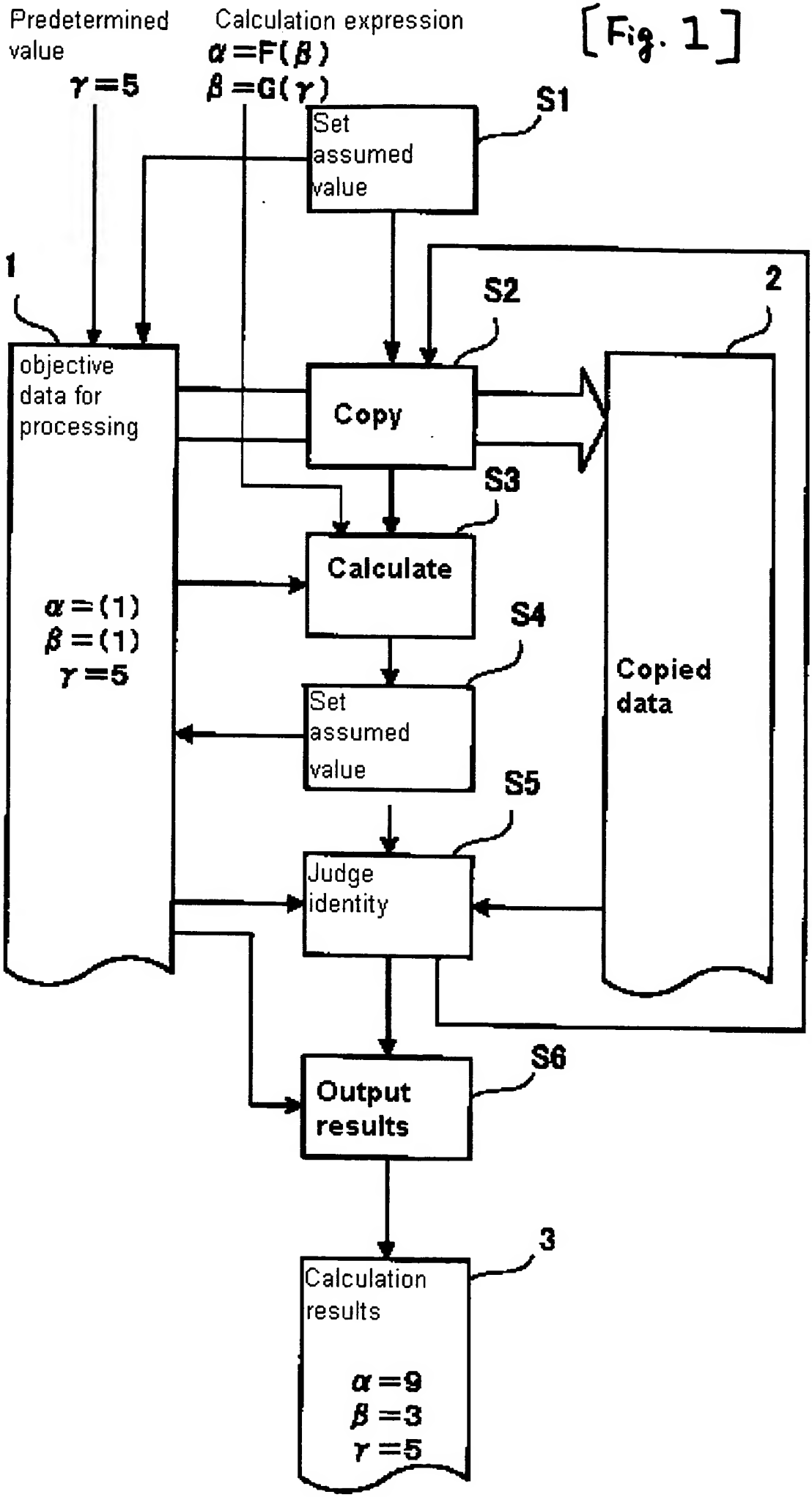
2 Copy Data

3 Result of an Operation

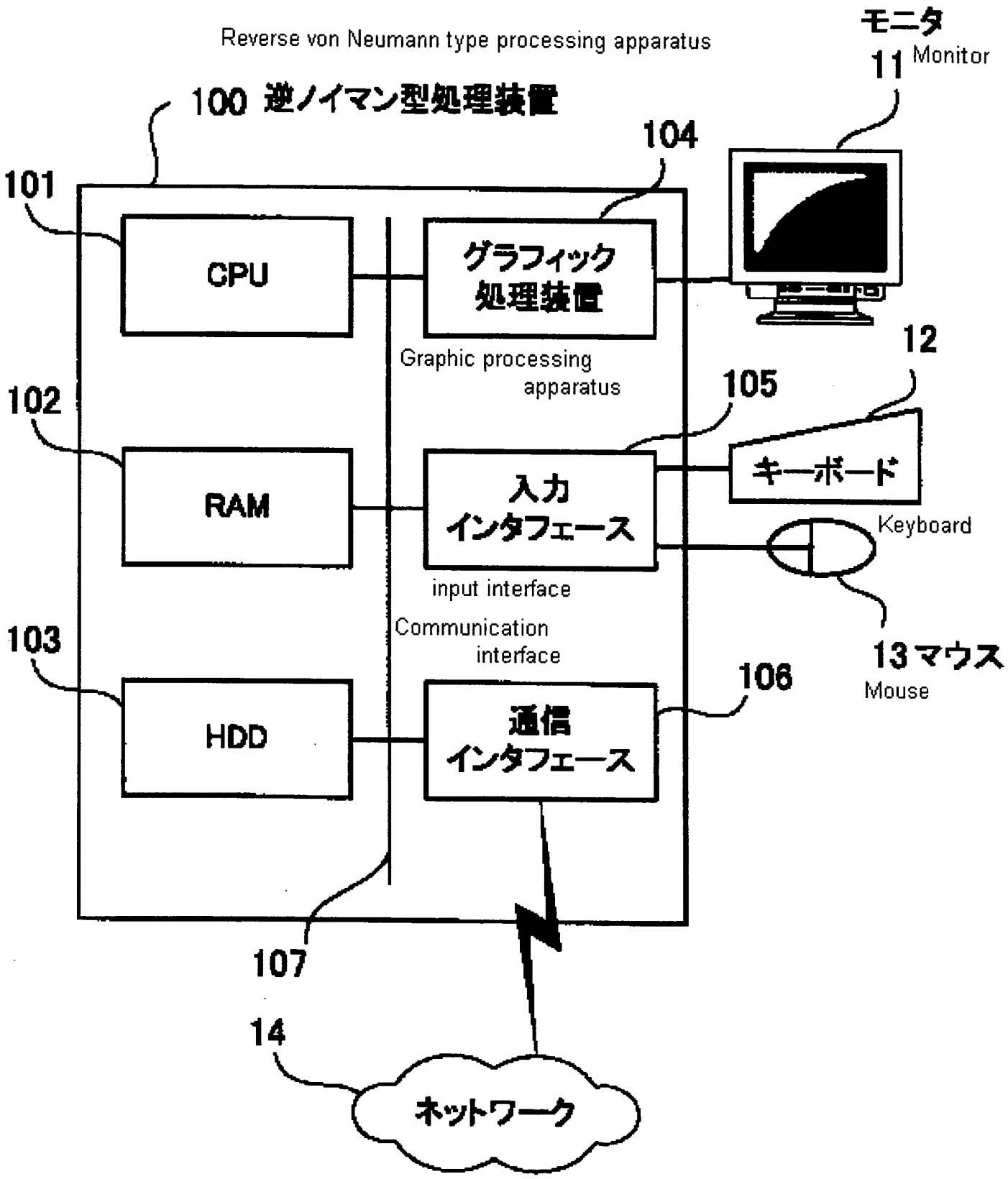
11 Monitor

12 Keyboard
13 Mouse
14 Network
100 Reverse J. Von Neumann Mold Processor
101 CPU
102 RAM
103 Hard Disk Drive Unit (HDD)
104 Graphics Processing Unit
105 Input Interface
106 Communication Interface
107 Bus

[Fig. 1]


Predetermined value
$\gamma = 5$
Calculation expression
$\alpha = F(\beta)$
$\beta = G(\gamma)$
S1
Set assumed value
1
S2
2
objective data for processing
Copy
S3
Calculate
$\alpha = (1)$
$\beta = (1)$
$\gamma = 5$
S4
Copied data
Set assumed value
S5
Judge identity
S6
Output results
3
Calculation results
$\alpha = 9$
$\beta = 3$
$\gamma = 5$

[Fig. 2]


Reverse von Neumann type processing apparatus
100 逆ノイマン型処理装置
モニタ
11 Monitor
101
104
CPU
グラフィック
処理装置
Graphic processing
apparatus
12
102
105
RAM
入力
インタフェース
キーボード
Keyboard
input interface
Communication
interface
103
106
13 マウス
Mouse
HDD
通信
インタフェース
107
14
ネットワーク